M. J. Flynn J. N. Gray A. K. Jones K. Lagally
H. Opderbeck G. J. Popek B. Randell
J. H. Saltzer H. R. Wiehle

Operating Systems

An Advanced Course

Edited by
R. Bayer, R. M. Graham, and G. Seegmüller

Springer-Verlag Berlin Heidelberg New York

M. J. Flynn J. N. Gray A. K. Jones K. Lagally
H. Opderbeck G. J. Popek B. Randell
J. H. Saltzer H. R. Wiehle

Operating Systems

An Advanced Course

Edited by
R. Bayer, R. M. Graham, and G. Seegmüller

Springer-Verlag
Berlin Heidelberg New York 1979

PREFACE

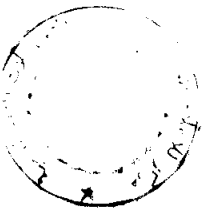
The Advanced Course on Operating Systems was held at the Technical University in Munich from July 28 to August 5, 1977, and was repeated from March 29 to April 6, 1978. The course was organized by the Institute for Informatics of the Technical University Munich and the Leibniz Computing Center of the Bavarian Academy of Sciences, in co-operation with the European Communities, sponsored by the Ministry for Research and Technology of the Federal Republic of Germany.

Contents

CHAPTER 1.: INTRODUCTION

CHAPTER 2.: MODELS

CHAPTER 3.: ISSUES AND RESULTS IN THE DESIGN OF OPERATING SYSTEMS



CHAPTER 1: INTRODUCTION

R. Bayer

Technical University Munich
Munich, Germany

R. M. Graham

University of Massachusetts
Amherst, Mass., USA

J. H. Saltzer

Massachusetts Institute of Technology
Cambridge, Mass., USA

G. Seegmüller

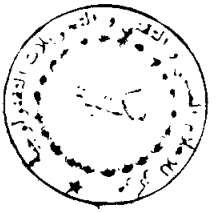
Leibniz Computing Center
of the Bavarian Academy of Sciences
Munich, Germany

Programming systems consisting of

editors, compilers, debuggers, ...
the operating system,
the hardware.

Data base systems consisting of

data base managers,
the operating system,
the hardware.



Application systems consisting of

application programs,
the operating system,
the hardware.

There is also agreement on those aspects that are at the heart of operating systems. In fact, the terms nucleus or kernel are often used for the most essential functions of an operating system. Much of the research and development in operating systems has focused on resource management and the user's interface to this management. Our view of operating systems and the focus of this course is resource management in a very wide sense and the attendant user interface. We shall concentrate on the semantics of this interface, on internal system structure and, to some extent, on hardware architecture.

It is interesting and instructive to look briefly at the history of modern computer systems. In the beginning, computers were small, simple, and free standing. Each individual could use the machine on a one-to-one basis. Generally, there has been an evolution from this state to the current large, complex, multiprogramming, multiprocessor, central systems with virtual memory and many ancillary devices and subsystems. The major trends have been: from one user to many users of the same system; from isolated users to cooperating users; from sequential batch to multiprogramming, to time sharing; and, in both hardware and software, an increase in the degree of concurrency. Most importantly, we see a trend toward increased concern with the management of non-physical resources.

The first computer users always had the entire computer all to themselves for some interval of time. A user always had all the resources. Any resource management facilities provided by an operating (or programming) system were entirely for the user's convenience. As the user community grew it was necessary to insure efficient,

the problem to be solved. The notion of an abstract machine which is available to each user encompasses the essence of this direction of abstraction.

What is the current state of affairs? In a recent workshop the lecturers of this course concluded that the classic problems of physical resource management and concurrency management are well understood, at least to the extent that their implementation is routine and minor enough that operating systems that are satisfactory to the market place are being built. We have chosen to omit from this course any consideration of these problems. Acceptable solutions are widely known. In fact, all of the recent textbooks on operating systems contain extensive discussions of these problems and their solutions. Rather we tried to focus on problems that were less well understood in the past - that are on or near the frontier of the field and that showed significant progress within the last few years. For example, none of the textbooks has an adequate discussion of protection, yet this is one of the most important problems in the design of new operating systems.

Abstractions are based on models. We recognize that models are not only needed to cope with complexity, but ultimately they are needed to verify or validate the correctness and other desired properties of a specific system design. Models for the underlying hardware are the foundation upon which more abstract, general models are built, since they give us insight into the fundamental mechanisms for the final interpretation of a program that is required to produce actual results. In addition, through them we can glimpse a future kind of architecture with many parallel activities, highly distributed.

The object model is the basis for the abstract resource, an object. This very general model is applicable to both software and hardware. It has benefitted from more recent developments in the study of programming languages. This benefit is not incidental. There, the need for careful specification of interfaces with total protection of their implementation has led to the introduction of abstract data types. Objects in operating systems correspond to data types as they appear in some more recent programming languages. The object model seems, in some sense, to capture fundamental properties that pervade all aspects of modern operating systems: protection, naming, binding, data, procedures, and physical devices. A model of this nature seems to be necessary in order to realistically consider the validation of important properties of an operating system, such as correctness and reliability.

There are a substantial number of major problems that affect the entire fiber of the more advanced operating systems. Most of these problems appear in the newer system organizations, such as, data base operating systems, distributed systems, and networks of computers. In these new settings the problems tend to be an order of magni-